ภก.สำราญ โป๊ะสูงเนิน

SOFT MAIL

เทคนิการสร้าง ปฏิทิน ฉบับพิเศษ

สร้างปฏิทินที่สามารถแสดงได้ถึง 500 ปี และพกพาไปที่ไหน ๆ ก็สะดวกสบาย...

เนื่องจากมีวันหยุดติดต่อกันหลายวัน ทำให้ได้มีเวลาได้มาสนุกแกมเครียดกับศึกเทพเจ้า (game GODS) ที่เล่นค้างไว้จากหลายวันก่อน ขณะเพลิน ๆ อยู่นั้นเอง เพื่อนผู้ไม่ได้พบกันมาเกือบ 5 ปีก็โผล่มาหา หลังไต่ถามสารทุกข์สุกดิบกันพอหอมปากหอมคอแล้ว เพื่อนก็ตั้งคำถามกับนักคอมพิวเตอร์มือใหม่ว่า

1. เครื่องคอมพิวเตอร์สามารถสร้างปฏิทินได้หรือไม่ (ตอบว่าสร้างได้สบายมาก)

2. ถ้าเป็นปฏิทินที่แสดงได้สัก 400-500 ปีล่ะ (อย่าว่าแต่ 500 ปีเลย เป็นล้าน ๆ ปีก็ได้)

3. การใช้ปฏิทินนี้ผู้ใช้ไม่ต้องคำนวณ ไม่ต้องป้อนตัวเลขอะไรเลย ดูอย่างเดียว

4. ให้พิมพ์ปฏิทินที่มีคุณสมบัติดังกล่าวนี้ออกมาอยู่ในกระดาษเพียงแผ่นเดียว เพื่อจะถือไปใช้งานที่ใด ๆ โดยไม่ต้องหิ้วเครื่องไปด้วย ได้หรือเปล่า

เริ่มต้นแก้ปัญหา

ความจริงหลังจากจบคำถามทั้ง 4 แล้ว ผมก็ได้แนะนำว่าไปหาซื้อปฏิทินโหราศาสตร์ 500 ปีมาก็ได้หรือถ้ามีเงินหน่อยก็ซื้อคอมพิวเตอร์โน้ตบุ๊คมาวิ่งโปรแกรมสร้างปฏิทินอย่าง Calendar Creator Plus หรือพวกยูทิลิตี้ เช่น Side Kick หรือพีซีทูลส์ก็ได้ แต่สุดท้ายก็พบว่าเรายังไม่ได้แก้ปัญหาทั้ง 4 ข้อนี้เลย กล่าวคือ

1. แม้โปรแกรมสำเร็จรูปเหล่านั้นจะสามารถสร้างปฏิทินได้ก็จริง แต่สร้างได้ไม่มากปีนัก นั่นคือมีข้อจำกัดของตัวโปรแกรมเอง เช่น Side Kick รุ่น 1.56A จะแสดงปฏิทินได้จากปี พ.ศ. 2444-พ.ศ. 2642 (ค.ศ. 1901-2099) ส่วนโปรแกรม Calendar Creator Plus 4.0 สร้างปฏิทินได้ในช่วงปี พ.ศ. 2343-พ.ศ. 2643 (ค.ศ. 1800-2100) เท่านั้น

2. ผู้ใช้ยังคงต้องป้อนปี พ.ศ., ปี ค.ศ. ลงไปในเครื่องหรืออาจต้องกระทำการอีกหลายอย่าง กว่าที่จะได้ปฏิทินปีนั้น ๆ ออกมา รวมทั้งต้องมาศึกษาโปรแกรมสำเร็จรูปเหล่านั้นเสียก่อน จึงจะใช้งานได้

3. ยังต้องหิ้วเครื่องไปไหนมาไหนด้วย และถ้าพิมพ์ออกมาก็จะได้ปฏิทินเพียงปีเดียว หลายปีก็หลายแผ่น

4. ถ้าซื้อหนังสือโหราศาสตร์ ก็จะได้หนังสือเล่มใหญ่หนาปึ้กเปิดหาไม่สะดวก พกพาลำบาก และข้อสุดท้ายที่สำคัญที่สุดก็คือเสียเหลี่ยมนักเลง ดังนั้นผมจึงขอเวลาสำหรับสร้างปฏิทินฉบับพิเศษนี้ 1 วัน

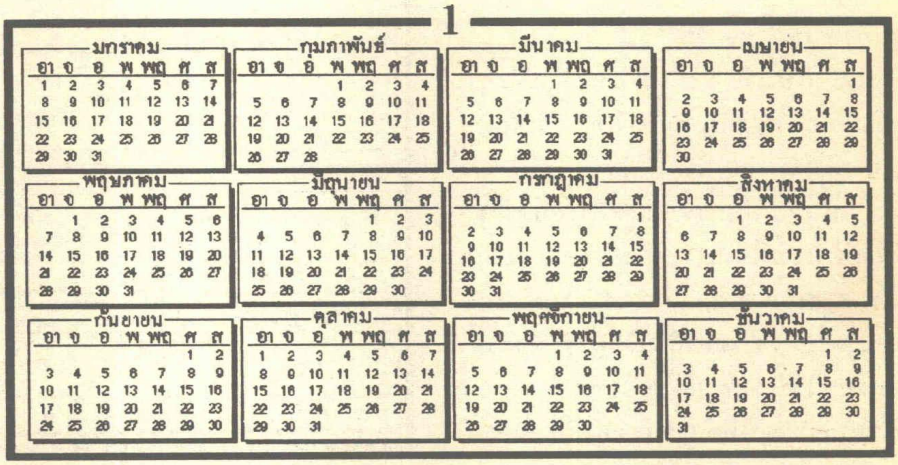
เมื่อมาพิจารณาถึงข้อกำหนดทั้ง 4 ข้อ จะพบว่าข้อที่ 1 และ 2 ไม่มีปัญหาเครื่องทำได้แน่ และมีผู้ทำไว้หลายรายแล้ว เช่น ไดอารีหนึ่งพันปี และปฏิทินในช่วงพันล้านปี (ของฝาก) ในไมโครคอมพิวเตอร์ฉบับที่ 65 เดือนธันวาคม 2533 (มีซอฟต์เมล์ด้วย) และคิดว่านักเขียนโปรแกรมทั้งหลายคงสร้างได้เพราะเป็นพื้นฐานจริง ๆ ที่นักเรียนซึ่งเรียนภาษาปาสคาล ซีหรือฟอร์แทรนต้องถูกอาจารย์สั่งให้เขียนโปรแกรมสร้างปฏิทินไว้ใช้งาน ดังนั้นผมจึงไม่ขอแสดงโปรแกรมสร้างปฏิทินในบทความนี้

ในข้อกำหนดที่ 3 หากไม่ต้องการให้ผู้ใช้คำนวณหรือทำอะไรเลย ผู้สร้างปฏิทินคงต้องคำนวณหรือสร้างปฏิทินที่เป็นไปได้ทั้งหมดในช่วง 500 ปีนี้ไว้ให้ผู้ใช้ ซึ่งข้อนี้จะสัมพันธ์กับข้อกำหนดที่ 4 เพราะหากสร้างไว้ใหญ่โตเกินไปก็ไม่สามารถพิมพ์ปฏิทินทั้งหมดลงในหน้ากระดาษที่ถูกจำกัดไว้ได้

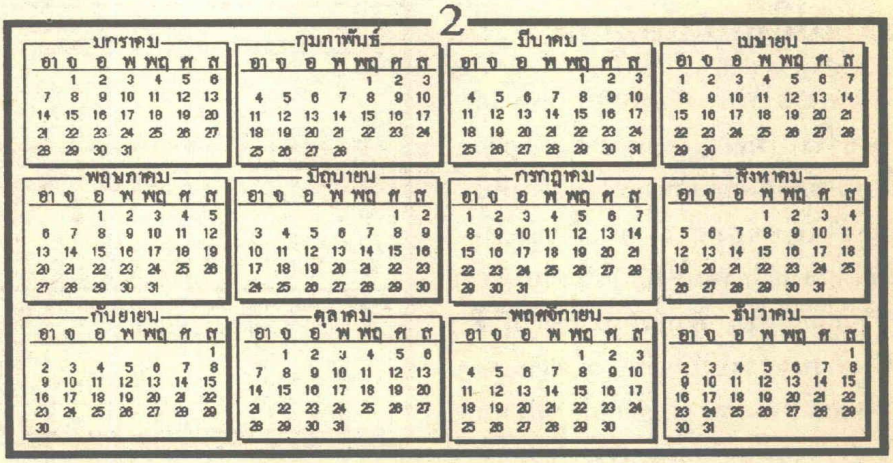
หลักการสำคัญ ๆ

โดยหลักความจริงแล้ว รูปแบบของสุริยะปฏิทินภายใน 500 ปีหรือมากกว่านั้น จะวนเวียนซ้ำกันไปมาหรือมีโอกาสเป็นไปได้อยู่ภายใน 14 แบบเท่านั้น ทั้งนี้เพราะวันที่ 1 มกราคมของทุกปีมีโอกาสตรงกับวันต่าง ๆ ได้ 7 วัน คือวันอาทิตย์ จันทร์ อังคาร พุธ พฤหัสบดี ศุกร์ และเสาร์ วันใดวันหนึ่งเท่านั้น และจำนวนวันทั้งหมดในแต่ละปีก็เป็นได้เพียง 365 วันหรือ 366 วันเท่านั้น ซึ่งดูได้จากเดือนกุมภาพันธ์มี 28 หรือ 29 วันนั่นเอง ดังนั้น ความเป็นไปได้ทั้งหมดจะเท่ากับ 7×2 = 14 กรณีกล่าวคือ

- แบบที่ 1 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันอาทิตย์ และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 2 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันจันทร์ และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 3 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันอังคาร และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 4 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันพุธ และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 5 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันพฤหัสบดีและเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 6 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันศุกร์ และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 7 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันเสาร์ และเดือนกุมภาพันธ์มี 28 วัน
- แบบที่ 8 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันอาทิตย์ และเดือนกุมภาพันธ์มี 29 วัน



รูปที่ 1 แสดงปฏิทินแบบที่ 1



รูปที่ 2 แสดงปฏิทินแบบที่ 2

- แบบที่ 9 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันจันทร์ และเดือนกุมภาพันธ์มี 29 วัน

- แบบที่ 10 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันอังคาร และเดือนกุมภาพันธ์มี 29 วัน

- แบบที่ 11 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันพุธ และเดือนกุมภาพันธ์มี 29 วัน

- แบบที่ 12 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันพฤหัสบดี และเดือนกุมภาพันธ์มี 29 วัน

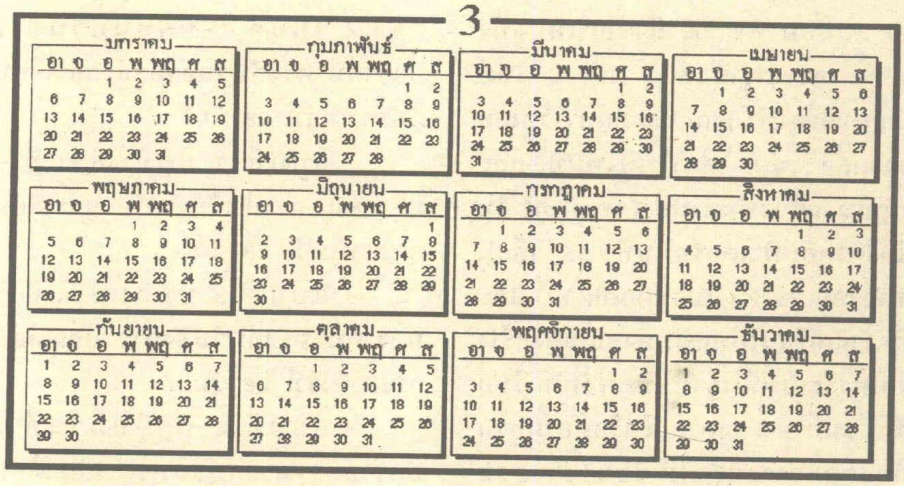
- แบบที่ 13 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันศุกร์ และเดือนกุมภาพันธ์มี 29 วัน

- แบบที่ 14 ปฏิทินมีวันที่ 1 มกราคม ตรงกับวันเสาร์ และเดือนกุมภาพันธ์มี 29 วัน

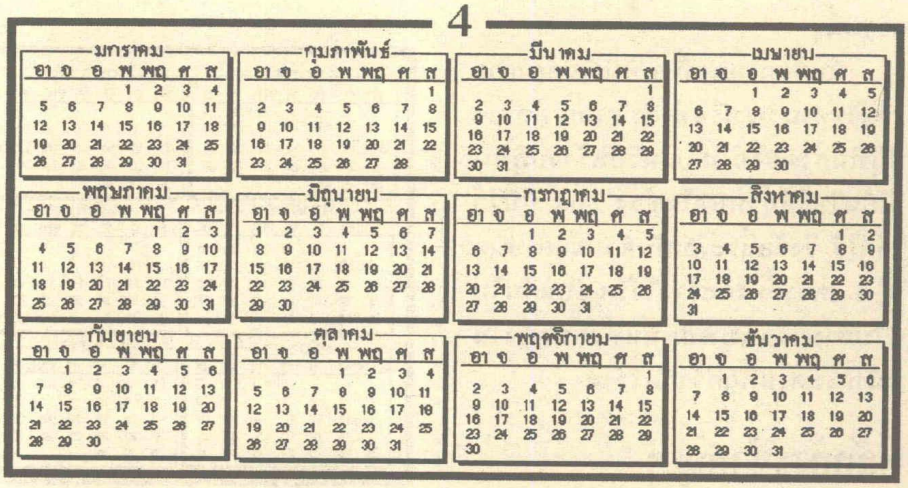
นั่นคือ ต้องสร้างและพิมพ์ปฏิทินทั้ง 14 แบบนี้ขึ้นมา แล้วคำนวณหาว่าปี พ.ศ. หรือ ค.ศ. ใดจะตรงกับปฏิทินแบบใดใน 14 แบบนี้ไว้ให้ผู้ใช้ ซึ่งผมขอเรียกวิธีนี้ว่าการใช้ระบบดัชนี โดยผู้ใช้เพียงแต่ดูปีที่ต้องการว่ามีหมายเลขดัชนีตรงกับแบบใดแล้วไปใช้ปฏิทินตามแบบนั้น ๆ

สร้างปฏิทินฉบับพิเศษ

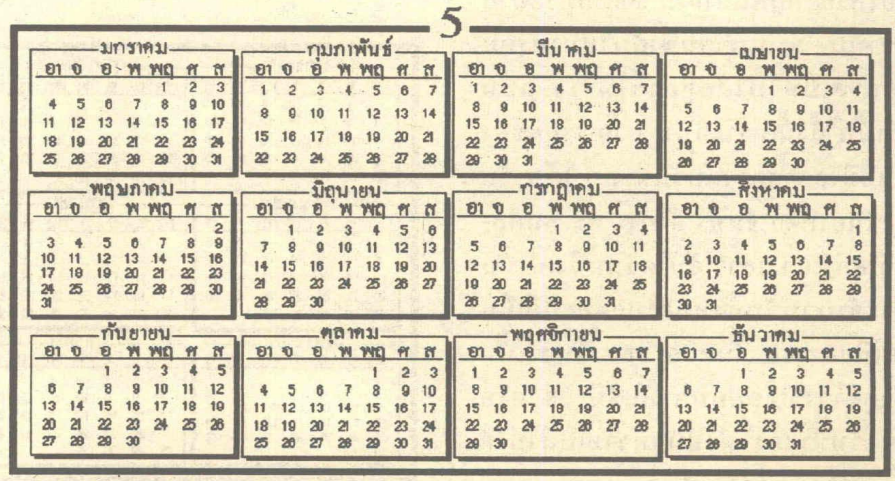
การสร้างและพิมพ์ปฏิทินนี้ผมได้ลองสร้างโปรแกรมขึ้นมา แต่ปรากฏว่าเมื่อพิมพ์ผลลัพธ์ออกมาจะได้ปฏิทินที่มีขนาดใหญ่พอสมควร แม้ว่าจะปรับฟอนต์ของเครื่องพิมพ์ให้เป็นตัวบีบหรือตัวจิ๋วแล้วก็ตาม ดังนั้นผมจึงหันไปใช้โปรแกรมสร้างปฏิทินคือ Calendar Creator Plus สร้างปฏิทินแบบต่าง ๆ ขึ้นมา จากนั้นเก็บปฏิทินไว้ในรูปแบบของ PCX ไฟล์ เพื่อใช้โปรแกรมตกแต่ง



รูปที่ 3 แสดงปฏิทินแบบที่ 3



รูปที่ 4 แสดงปฏิทินแบบที่ 4



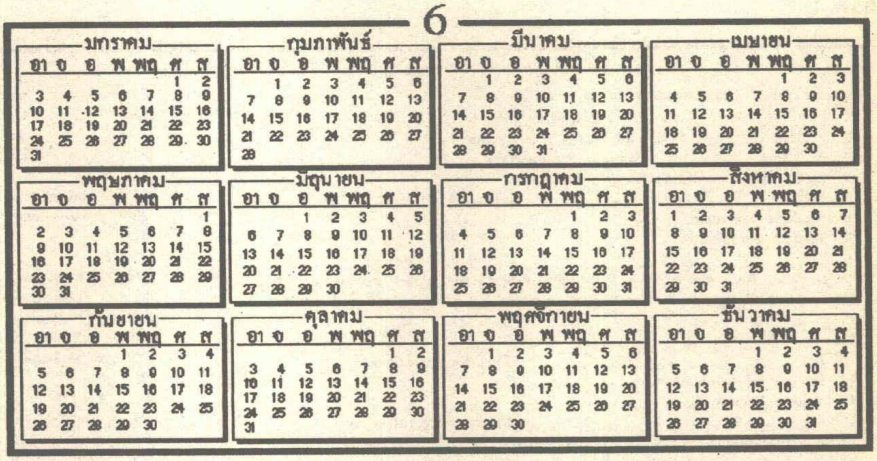
รูปที่ 5 แสดงปฏิทินแบบที่ 5

รูปภาพทั้งหลาย เช่น เพนต์บรัชของวินโดวส์ CorelDraw ฯลฯ ย่อหรือขยาย ใส่ภาษาไทยหรือลูกเล่นต่าง ๆ ลงในปฏิทินได้อย่างสวยงาม

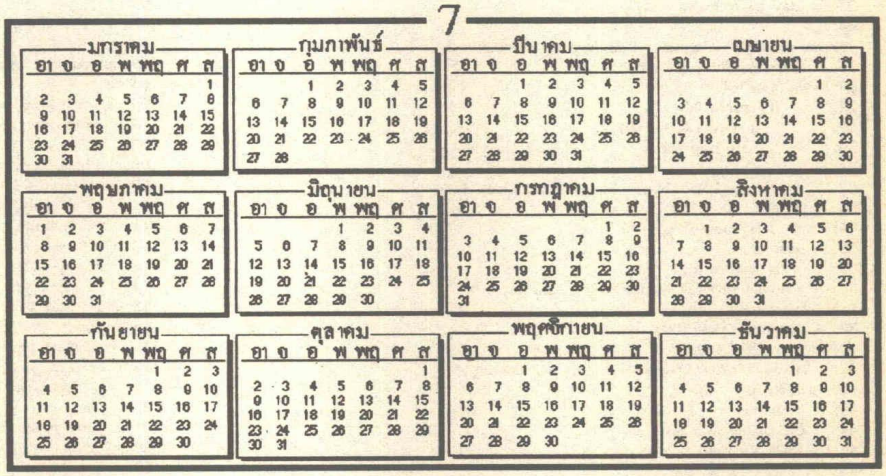
อนึ่งในการสร้างนั้น เราสร้างปฏิทินเพียง 7 แบบก่อนก็ได้ ส่วนแบบที่ 8 ถึง 14 นั้นใช้วิธีการลอกจากแบบที่สร้างไว้ก่อนหน้านั้น โดยเปลี่ยนแปลงเฉพาะเดือนมกราคมและกุมภาพันธ์เท่านั้น เช่น ปฏิทินแบบที่ 8 จะใช้โครงสร้างเดือนมีนาคมถึงธันวาคมของแบบที่ 2 แล้วนำเดือนมกราคมและกุมภาพันธ์จากแบบที่ 1 มาตัดต่อโดยเพิ่มเติมวันที่ 29 กุมภาพันธ์เข้าไปด้วย ซึ่งช่วยให้สร้างปฏิทินทั้ง 14 แบบได้เร็วขึ้น

ส่วนโปรแกรมสร้างดัชนีนั้น ผมเขียนด้วย Turbo C++ 1.0 ซึ่งจริง ๆ แล้วจะใช้ภาษาอะไรก็ได้ ไม่ว่าจะเป็นภาษาเบสิก ปาสคาล หรือแม้แต่โปรแกรมด้วยภาษาดีเบส ผมเพียงแต่อยากจะลองของใหม่เท่านั้นเอง และเขียนอย่างซีธรรมดา ไม่ใช่โอโอพีอะไรที่ไหนเลย

หลักการทำงานมีอยู่ว่าโปรแกรมจะรับค่าปี พ.ศ. จากคอมมานด์ไลน์ โดยรับเป็นช่วงปีจากปี พ.ศ. หนึ่ง ถึงอีก พ.ศ. หนึ่ง เช่น ปี พ.ศ. 2501 ถึง พ.ศ. 3000 เป็นต้น แล้วนำไปตรวจสอบว่าแต่ละปีนั้นมีปีใดบ้างที่เป็นปีอธิกสุรทิน จากนั้นคำนวณหาวันที่ 1 มกราคมของแต่ละปีว่าตรงกับวันอะไร เมื่อได้ข้อมูลทั้งสองกรณีนี้แล้ว จะสร้างดัชนีตามเงื่อนไขใน 1-14 แบบให้ โดยจะพิมพ์ผลลัพธ์ออกมาในแนวคอลัมน์ ซึ่งคิดว่าจะสะดวกต่อการค้นหาปี พ.ศ. มากกว่าการกวาดสายตาแบบซ้ายไปขวาต่อด้วยบนลงล่าง ดังมีรายละเอียดในลิสติงที่ 1 ซึ่งจะอธิบายได้ดังนี้



รูปที่ 6 แสดงปฏิทินแบบที่ 6



รูปที่ 7 แสดงปฏิทินแบบที่ 7

บรรทัดที่ 9-12 กำหนดค่าคงที่ให้เริ่มจากปี พ.ศ. 2501-3000 รวมเป็นจำนวน 500 ปี เพื่อใช้เป็นค่าดีฟอลต์ ซึ่งท่านสามารถแก้ไขให้เป็นปีใด ๆ ก็ได้ที่อยู่ในช่วง 544 ถึง 32767

บรรทัดที่ 23 จะเริ่มฟังก์ชัน day__of__week เพื่อหาค่าดัชนีของแต่ละปี โดยในบรรทัดที่ 27-36 ใช้ตรวจสอบว่าปีที่รับเข้ามาเป็นปีอธิกมาสหรือไม่ จะเห็นว่านอกจากจะตรวจสอบว่าปี ค.ศ. ใดหารด้วย 4 ลงตัวแล้ว ยังตรวจสอบทุก ๆ 100 ปีด้วย หากปีนั้นเป็นปีที่ครบศตวรรษแต่หารด้วย 400 ไม่ลงตัว ก็ไม่ถือว่าเป็นปีอธิกมาส แม้ว่าจะหารด้วย 4 ลงตัวก็ตาม เช่น ปี ค.ศ. 1800 ค.ศ. 1900 เป็นต้น ทั้งนี้เพื่อจะได้เป็นมาตรฐานเดียวกันกับโปรแกรมซึ่งผมใช้สร้างแบบปฏิทิน คือ Calendar Creator Plus นอกจากจะหาปีอธิกมาสแล้ว ฟังก์ชันนี้ยังหาว่าวันที่ 1 มกราคมตรงกับวันอะไรด้วย โดยใช้สูตรในบรรทัดที่ 41-43 ซึ่งดัดแปลงมาจาก Zeller's Congruence ค่าที่ได้จะเป็นเลข 0 ถึง 6 ที่หมายถึงวันแต่ละวัน กล่าวคือ เลข 0

ลิสติงที่ 1

[Source code of Calendar.cpp]

```
1: //_____________________ Calendar.CPP
2: //  Use..... Create the indexs of any years
3: //  Name..... Mr. Sumran  Posungnoen
4: //  Compiler..... Turbo C++ 1.0
5: //_____________________________Including file
6: #include <iostream.h>
7: #include <stdlib.h>
8: //_____________________________Constant declaration
9: const int Start_year = 2501,  // Buddhist Era(B.E.) = 1958 Christian Era(C.E.)
10:         End_year   = 3000,  // Buddhist Era(B.E.) = 2457 Christian Era(C.E.)
11:         Yes = 1,
12:         No  = 0;
13: //_____________________________Function prototype
14: int  day_of_week(int year);
15: void to_help(void);
16: //_____________________________Function day_of_week
17: //   Function name : day_of_week(int year)
18: //   Use : 1. Check the leap-year
19: //           2. Find the day of the first week of the year
20: //           .3. Ceate an index
21: //   Parameters : (integer) year
22: //   Return : index of each year
23: int day_of_week(int year)
24: {
25:  int century = 0, n_day = 0, is_leap = No, index = 0;
26: //  Checking for leap year
27:    if (year%4 == 0)
28:       if (year%100 == 0)
29:          if (year%400 == 0)
30:             is_leap = Yes;          // It is a leap-year
31:          else
32:             is_leap = No;           // It is not a leap_year
33:       else
34:          is_leap = Yes;
35:    else
36:       is_leap = No;
37: //  Find day of week; Sunday = 0 .. Saturday = 6
38:    year  -= 1;
39:    century = year/100;
40:    year  %= 100;
41:    n_day = (29+(5*year/4)+century/4-2*century)%7;
42:    if (n_day < 0)
43:       n_day  += 7;
44: //  Caiculated an index
45:    switch(n_day) {
46:     case 0 : if(is_leap)
47:                 index = 8;
48:              else
49:                 index = 1;
50:              break;
51:     case 1 : if(is_leap)
52:                 index = 9;
53:              else
54:                 index = 2;
55:              break;
56:     case 2 : if(is_leap)
57:                 index = 10;
58:              else
59:                 index = 3;
60:              break;
61:     case 3 : if(is_leap)
62:                 index = 11;
63:              else
64:                 index = 4;
65:              break;
66:     case 4 : if(is_leap)
67:                 index = 12;
68:              else
69:                 index = 5;
70:              break;
71:     case 5 : if(is_leap)
72:                 index = 13;
73:              else
74:                 index = 6;
75:              break;
76:     case 6 : if(is_leap)
77:                 index = 14;
78:              else
79:                 index = 7;
80:              break;
81:    }
82:    return(index);
83: } //--------- end of day_of_week function
84:
85: //_____________________________Function to_help
86: //   Function name : to_help(void)
87: //   Use : Show help
88: //   Parameters : none
89: //   Return : none
90: void to_help(void)
91: {
92:   cout << "\n   Usage: Calendar [Start_year] [End_year] \n";
93:   cout << "\n   Parameter : The value of the Start_year or the End_year is";
94:   cout << "\n               the numberic between 544 to 32767 B.E.";
95:   cout << "\n   Default : Start_year = 2501 B.E. (1958 C.E.)";
96:   cout << "\n             End_year   = 3000 B.E. (2457 C.E.)";
97:   cout << "\n   Example : Calendar 2536 2600";
98:   cout << "\n             This program will print the indexs from the year";
99:   cout << "\n             2536 B.E. to the year 2600 B.E.\n";
100: }
101: //--------- end of to_help function
102:
103: int main(int argc, char *argv[])
104: {
105:  int first_year, last_year, any_year,
106:     step, row, col, stop = No;
107:
108: // Check input parameters
109:    if (argc < 2){
110:       to_help();
111:       return(1);
112:    }
113:    else{
114:       first_year = atoi(argv[1])-543;   // Convert to C.E. for calculating
115:       last_year  = atoi(argv[2])-543;
116:    }
117:    if (first_year < 1 || last_year < 1){
118:       first_year = Start_year-543;
119:       last_year  = End_year-543;
120:    }
121:    any_year = first_year;
122:    cout << "The indexs of the years from " << first_year+543 << " B.E. to "
123:         << last_year+543 << " B.E. are...\n";
124:    while (any_year <= last_year){
125:          step = 0;
126:          for (col = 0; col < 10 ; col++){
127:             if (any_year+step > last_year){
128:                stop = Yes;
129:                break;
130:             }
131:             else{
132:                cout.width(4);
133:                cout << any_year+step+543 << "...";
134:                cout.width(2);
135:                cout << day_of_week(any_year+step);
136:                cout << " ";
137:             }
138:             step  +=  50;
139:          }
140:          cout << "\n";
141:          row++;
142:          if (row%50 == 0){
143:             if (stop)
144:                return(0);
145:             else
146:                any_year = any_year+step-50;
147:             cout  << "\r\n";
148:          }
149:          any_year++;
150:    }
151:    return(0);
152: } // end of function main
153: //_____________________ End of file Calendar.CPP
```

```
Usage: Calendar [Start_year] [End_year]
Parameter : The value of the Start_year or the
      End_year is the numberic between 544 to 32767
      B.E.
Default : Start_year = 2501 B.E. (1958 C.E.)
      End_year = 3000 B.E. (2457 C.E.)
Example : Calendar 256 2600
      This program will print the indexs from the year
      2536 B.E. to the year 2600 B.E.
```

รูปที่ 8 แสดงหน้าจอเมื่อเรียกไฟล์ Help ขึ้นมาดู

หมายถึงวันปีใหม่ตรงกับวันอาทิตย์ เลข 2 ตรงกับวันจันทร์...จนถึงเลข 6 คือวันเสาร์ เมื่อทราบว่าเป็นปีอธิกมาสหรือไม่ และวันที่ 1 มกราคมตรงกับวันใด ก็สามารถให้ดัชนีของแต่ละปีได้ดังคำสั่งในบรรทัดที่ 45-82

บรรทัดที่ 85-101 เป็นฟังก์ชันสำหรับให้ความช่วยเหลือในการใช้งานโปรแกรมสร้างดัชนีนี้ หากท่านไม่ป้อนค่าพารามิเตอร์ใด ๆ โปรแกรมจะแสดง Help นี้ออกมา แต่ถ้าป้อนพารามิเตอร์ไม่ครบหรือป้อนไม่ถูกต้องเช่น ไม่ป้อนปี พ.ศ. เริ่มต้น ปี พ.ศ. สุดท้าย หรือป้อนแต่ไม่อยู่ในช่วงที่กำหนด โปรแกรมจะใช้ค่าดีฟอลต์ในการคำนวณปีที่ป้อนให้โปรแกรมนี้จะเป็นปี พ.ศ. เสมอ เพื่อความเป็นไทยน่ะครับ

ในฟังก์ชันเมนนั้นส่วนสำคัญจะเป็นการพิมพ์ผลลัพธ์ บรรทัดที่ 126-139 จะเป็นการสั่งให้พิมพ์ออกมาได้สูงสุดไม่เกิน 10 คอลัมน์ต่อบรรทัด แต่ละคอลัมน์จะห่างกัน 50 ปี ส่วนบรรทัดที่ 141-148 ใช้ตรวจสอบว่าหมดหนึ่งหน้าแล้วหรือไม่ โดยกำหนดให้มีหน้าละ 50 บรรทัด ผลลัพธ์จะถูกพิมพ์ออกมาทางจอภาพเท่านั้น ดังนั้นหากท่านต้องการพิมพ์เก็บไว้ในแฟ้มข้อมูลให้ใช้วิธี Redirection เช่น เมื่อพิมพ์ว่า

```
calendar >help
```

จะเป็นการพิมพ์เฉพาะคำชี้แจงการใช้งานลงในไฟล์ชื่อ help เมื่อเรียกไฟล์ help ขึ้นมาดู จะเห็นเป็นดังรูปที่ 8

หรือพิมพ์ calendar 2501 2800 >output จะได้ผลลัพธ์จากปี พ.ศ. 2501 ถึง พ.ศ. 2800 ลงในไฟล์ชื่อ output ดังรูปที่ 9

สู่ความสำเร็จ

ปัญหาข้อสุดท้ายที่ต้องแก้คือ การพิมพ์ดัชนีและปฏิทินทั้ง 14 แบบลงบนกระดาษ ผมลองบีบทั้งหมดลงบนกระดาษขนาด A4 ก็สามารถทำได้ตามข้อกำหนด แต่ตัวหนังสือจะเล็กมากมองเห็นได้ลำบาก แต่เพื่ออำนวยความสะดวกผมจึงเลือกที่จะเปลืองกระดาษเพิ่มขึ้นโดยพิมพ์ดัชนี 500 ปีไว้ 1 หน้าและปฏิทิน 2 แบบต่อหนึ่งหน้า รวมต้องใช้กระดาษ A4 ทั้งหมด 8 หน้า และเพื่อความสวยงามผมจึงตัดกระดาษโปสเตอร์สีให้มีขนาดเท่ากับกระดาษ A4 จำนวน 2 คู่แล้วพิมพ์ทั้งหมดลงไป เมื่อเย็บเล่มก็จะได้ปฏิทินเป็นฉบับพิเศษที่เหมาะมือ

```
The indexs of the years from 2501 B.E. to 2800 B.E. are...
2501... 4   2551...10   2601... 3   2651... 8   2701... 1   2751...13
2502... 5   2552... 5   2602... 4   2652... 3   2702... 2   2752... 1
2503...13   2553... 6   2603...12   2653... 4   2703...10   2753... 2
2504... 1   2554... 7   2604... 7   2654... 5   2704... 5   2754... 3
2505... 2   2555... 8   2605... 1   2655...13   2705... 6   2755...11

2546... 4   2596... 4   2646... 2   2696... 2   2746... 7   2796... 7
2547...12   2597... 5   2647...10   2697... 3   2747... 8   2797... 1
2548... 7   2598... 6   2648... 5   2698... 4   2748... 3   2798... 2
2549... 1   2599...14   2649... 6   2699...12   2749... 4   2799...10
2550... 2   2600... 2   2650... 7   2700... 7   2750... 5   2800... 5
```

รูปที่ 9